



Semantic Web Grundlagen

OWL 2 – Syntax und Semantik

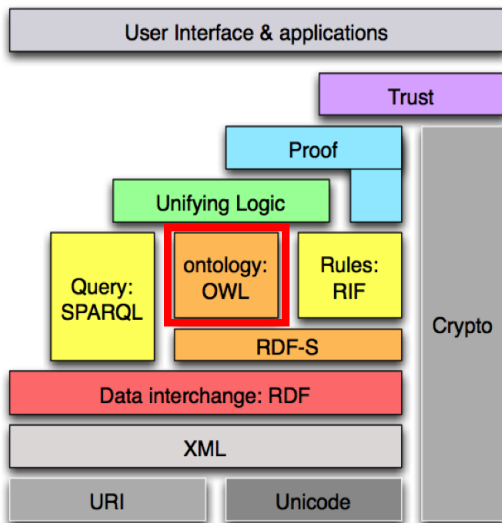
Birte Glimm
Institut für Künstliche Intelligenz | 24. Nov 2011

Foliensatz adaptiert von M. Krötzsch. Die nichtkommerzielle Vervielfältigung, Verbreitung und Bearbeitung ist zulässig (→ Lizenz CC-BY-NC).

Organisatorisches: Inhalt

Einleitung und XML	17. Okt	SPARQL Syntax	12. Dez
Einführung in RDF	20. Okt	Übung 4	15. Dez
RDF Schema	24. Okt	SPARQL Semantik	19. Dez
fällt aus	27. Okt	SPARQL 1.1	22. Dez
Logik – Grundlagen	31. Okt	Übung 5	9. Jan
Übung 1	3. Nov	SPARQL Entailment	12. Jan
Semantik von RDF(S)	7. Nov	SPARQL Implementierung	16. Jan
RDF(S) & Datalog Regeln	10. Nov	Abfragen & RIF	19. Jan
OWL Syntax & Intuition	14. Nov	Übung 6	23. Jan
Übung 2	17. Nov	Ontology Editing	26. Jan
OWL & BLs	21. Nov	Ontology Engineering	30. Jan
OWL 2	24. Nov	Linked Data	2. Feb
Tableau	28. Nov	Übung 7	6. Feb
Übung 3	1. Dez	SemWeb Anwendungen	9. Feb
Blocking & Unravelling	5. Dez	Wiederholung	13. Feb
Hypertableau	8. Dez	Übung 8	16. Feb

OWL 2



Agenda

- ▶ Rückblick: OWL & Überblick OWL 2
- ▶ Die Beschreibungslogik *SROIQ*
- ▶ Inferenz mit *SROIQ*
- ▶ OWL 2 DL
- ▶ OWL 2 Profiles
- ▶ OWL 2 Full
- ▶ Zusammenfassung

Rückblick: Erweiterungen von OWL

OWL für viele Aufgaben noch zu schwach

- ▶ OWL als Anfragesprache ungenügend
~> Konjunktive Anfragen, SPARQL für OWL
- ▶ OWL als Ontologiesprache ungenügend
~> Prädikatenlogische Regelerweiterungen, SWRL & RIF
- ▶ OWL als Programmiersprache ungenügend
~> Logikprogrammierung im Semantic Web

Sollte auch der OWL-Standard selbst erweitert werden?

~> OWL 2

Entwicklung von OWL 2

OWL 2 als “nächste Version” von OWL

Erweiterungen aufgrund von Praxiserfahrung mit OWL 1.0:

- ▶ zusätzliche Ausdrucksstärke durch neue ontologische Axiome
- ▶ nicht-logische Erweiterungen (Syntax, Metadaten, ...)
- ▶ Überarbeitung der OWL-Varianten (Lite/DL/Full)

Zielstellungen:

- ▶ weitestgehende Kompatibilität zum existierenden OWL-Standard
- ▶ Erhaltung der Entscheidbarkeit von OWL DL
- ▶ Behebung von Problemen im OWL 1.0 Standard

Agenda

- ▶ Rückblick: OWL & Überblick OWL 2
- ▶ Die Beschreibungslogik *SROIQ*
- ▶ Inferenz mit *SROIQ*
- ▶ OWL 2 DL
- ▶ OWL 2 Profiles
- ▶ OWL 2 Full
- ▶ Zusammenfassung

Von *SHOIN* zu *SROIQ*

OWL DL basiert auf Beschreibungslogik *SHOIN*(D):

- ▶ Axiome:
 - ▶ TBox: Subklassenbeziehungen $C \sqsubseteq D$
 - ▶ RBox: Subrollenbeziehungen $R \sqsubseteq S$ (\mathcal{H}), Inverse Rollen R^{-} (\mathcal{I}), Transitivität
 - ▶ ABox: Fakten zu Klassen $C(a)$, Rollen $R(a, b)$, und Gleichheit $a \approx b$ bzw. $a \neq b$
- ▶ Klassenkonstruktoren:
 - ▶ Konjunktion $C \sqcap D$, Disjunktion $C \sqcup D$, Negation $\neg C$ von Klassen
 - ▶ Rollenrestriktionen: universell $\forall R.C$ und existenziell $\exists R.C$
 - ▶ Zahlenrestriktionen (\mathcal{N}): $\leq n R$ und $\geq n R$ (n nicht-negative Zahl)
 - ▶ Nominale (\mathcal{O}): $\{a\}$
- ▶ Datentypen (D)

Erweiterung in OWL 2 zu *SROIQ*(D)

ABox

SHOIN unterstützt verschiedene ABox-Fakten:

- ▶ Klassenzugehörigkeit $C(a)$ (C komplexe Klasse),
- ▶ Sonderfall: negierte Klassenzugehörigkeit $\neg C(a)$ (C komplexe Klasse),
- ▶ Gleichheit $a \approx b$,
- ▶ Ungleichheit $a \not\approx b$
- ▶ Rollenbeziehungen $R(a, b)$
- ▶ *negierte Rollenbeziehungen?*

\rightsquigarrow *SROIQ* erlaubt auch **negierte Rollen** in der ABox: $\neg R(a, b)$

Zahlenrestriktionen

SHOIN unterstützt nur einfache Zahlenrestriktionen (\mathcal{N}):

Person $\sqcap \geq 3$ hatKind

„Klasse aller Personen mit 3 oder mehr Kindern.“ \rightsquigarrow *SROIQ*

erlaubt auch **qualifizierte Zahlenrestriktionen** (\mathcal{Q}):

Person $\sqcap \geq 3$ hatKind.(Frau \sqcap Professor)

„Klasse aller Personen mit 3 oder mehr Töchtern, die Professoren sind.“

Das Konzept Self

Modellierungsaufgabe: „Jeder Mensch kennt sich selbst.“

▶ *SHOIN*:

kennt(tom, tom) kennt(tina, tina) kennt(udo, udo) ...

↪ nicht allgemein anwendbar

▶ *SROIQ*: spezieller Ausdruck **Self**

Mensch $\sqsubseteq \exists \text{kennt}.\text{Self}$

Rollenaxiome in *SHOIN*

SHOIN bietet wenige Rollenaxiome:

- ▶ $\text{Trans}(r)$, `owl:TransitiveProperty`: r ist **transitiv**
Beispiel: $\text{Trans}(\text{liegtIn})$
- ▶ $\text{Sym}(r)$, `owl:SymmetricProperty`: r ist **symmetrisch**
Beispiel: $\text{Sym}(\text{verwandtMit})$
Auch $r \sqsubseteq r^-$
- ▶ $\text{Func}(r)$, `owl:FunctionalProperty`: r ist **funktional**
Beispiel: $\text{Func}(\text{hatVater})$
Auch $\top \sqsubseteq \leq 1r$
- ▶ $\text{InvFunc}(r)$, `owl:InverseFunctionalProperty`: r ist **invers funktional**
Beispiel: $\text{InvFunc}(\text{istVaterVon})$
Auch $\top \sqsubseteq \leq 1r^-$ oder $\text{Func}(r^-)$

Rollenaxiome in *SROIQ*

SROIQ bietet zusätzliche Aussagen über Rollen:

- ▶ $\text{Ref}(r)$, `owl:ReflexiveProperty`: r ist **reflexiv**,
 $(x, x) \in r^{\mathcal{I}}$ für alle Domänenindividuen x
Beispiel: $\text{Ref}(\text{kennt})$
- ▶ $\text{Irr}(r)$, `owl:IrreflexiveProperty`: r ist **irreflexiv**,
 $(x, x) \notin r^{\mathcal{I}}$ für alle Domänenindividuen x
Beispiel: $\text{Irr}(\text{hatKind})$
- ▶ $\text{Asym}(r)$, `owl:AsymmetricProperty`: r ist **asymmetrisch**,
 $(x, y) \in r^{\mathcal{I}}$ impliziert $(y, x) \notin r^{\mathcal{I}}$
Beispiel: $\text{Asym}(\text{hatKind})$
- ▶ $\text{Dis}(r, s)$, `owl:propertyDisjointWith`,
`owl:AllDisjointProperties`: r und s sind **disjunkt**,
 $(x, y) \notin r^{\mathcal{I}} \cap s^{\mathcal{I}}$ für alle x, y
Beispiel: $\text{Dis}(\text{hatVater}, \text{hatSohn})$

Die Universelle Rolle

*SR**OIQ* bietet zusätzliche eine universelle Rolle:

- ▶ **Universelle Rolle** U (`owl:TopObjectProperty`):
 $(x, y) \in U^I$ für alle x, y

Beispiel

$\top \sqsubseteq \leq 7\,000\,000\,000 U.Menschen$
(nicht empfohlen!)

- ↪ U ist vor allem als Gegenstück zu \top sinnvoll, z.B. als Wurzel der Rollenhierarchie in grafischen Editoren
- ▶ Gegenstück `owl:BottomObjectProperty` wurde auch eingeführt, hat aber kein Gegenstück in BLs
- ▶ Für Dataproperty analog `owl:TopDataProperty` und `owl:BottomDataProperty`

Komplexe Rolleninklusion

„Die Freunde meiner Freunde sind auch meine Freunde.“

↪ Kann in *SHOIN* ausgedrückt werden:
hatFreund ist transitiv.

„Die Feinde meiner Freunde sind auch meine Feinde.“

↪ Kann nicht in *SHOIN* ausgedrückt werden!

Komplexe Rolleninklusion

- ▶ RBox-Ausdrücke der Form $r_1 \circ r_2 \circ \dots \circ r_n \sqsubseteq s$
- ▶ Semantik: $(x_0, x_1) \in r_1^{\mathcal{I}}, (x_1, x_2) \in r_2^{\mathcal{I}}, \dots, (x_{n-1}, x_n) \in r_n^{\mathcal{I}}$,
impliziert $(x_0, x_n) \in s^{\mathcal{I}}$

Komplexe Rolleninklusion – Beispiele

Beispiel

$\text{hatFreund} \circ \text{hatFeind} \sqsubseteq \text{hatFeind}$:

wenn $(x, y) \in \text{hatFreund}^{\mathcal{I}}$ und $(y, z) \in \text{hatFeind}^{\mathcal{I}}$,
dann gilt auch $(x, z) \in \text{hatFeind}^{\mathcal{I}}$

Weitere Beispiele

$\text{teilVon} \circ \text{gehört} \sqsubseteq \text{gehört}$

$\text{hatBruder} \circ \text{hatKind} \sqsubseteq \text{istOnkelVon}$

Ausdrucksstärke der Komplexen Rolleninklusion

Wie kompliziert ist die komplexe Rolleninklusion?

Mit RBoxen kann man formale Sprachen kodieren:

Grammatik für Sprache der Wörter $ab, aabb, aaabbbb, \dots$:

$$\begin{array}{ll} \mathbb{L} ::= ab & \text{wird zu RBox} \\ \mathbb{L} ::= aLb & \end{array} \quad \begin{array}{l} r_a \circ r_b \sqsubseteq \ell \\ r_a \circ \ell \circ r_b \sqsubseteq \ell \end{array} \rightsquigarrow$$

$\exists \ell. \top \neq \perp$ (“ $\exists \ell. \top$ notwendig nicht-leer”) bedeutet*:

„Es gibt eine Kette aus r_a und r_b , die zur Sprache gehört.“

$\rightsquigarrow \exists \ell_1. \exists \ell_2. \top \neq \perp$ für zwei kodierte Sprachen ℓ_1 und ℓ_2 bedeutet:

„Es gibt ein Wort, das zu ℓ_1 und zu ℓ_2 gehört.“

*) bei entsprechender TBox! Leider gilt: Leerheit der Überschneidung kontextfreier Sprachen ist unentscheidbar. \rightsquigarrow

OWL mit Rolleninklusionen ist unentscheidbar

Reguläre RBoxen

Kann man komplexe Rolleninklusion zwecks Entscheidbarkeit einschränken?

- ▶ RBoxen sind wie Grammatiken für kontextfreie formale Sprachen
- ▶ Überschneidungen von kontextfreien Sprachen problematisch
- ↪ Einschränkung auf reguläre Sprachen!

Regularitätsbedingung für RIAs

Um die Entscheidbarkeit der typischen Aufgaben des automatischen Schlussfolgerns zu garantieren müssen die Rolleninklusionen einer Wissensbasis **regulär** sein

- ▶ Es muss eine strikte lineare Ordnung \prec über die Rollen geben, so dass jedes RIA eine der folgenden Formen hat mit $s_i \prec r$ für alle $1 \leq i \leq n$:

$$\blacktriangleright r \circ r \sqsubseteq r$$

$$\blacktriangleright r \circ s_1 \circ s_2 \circ \dots \circ s_n \sqsubseteq r$$

$$\blacktriangleright r^- \sqsubseteq r$$

$$\blacktriangleright s_1 \circ s_2 \circ \dots \circ s_n \circ r \sqsubseteq r$$

$$\blacktriangleright s_1 \circ s_2 \circ \dots \circ s_n \sqsubseteq r$$

Regularitätsbedingung für RIAs

- ▶ **Beispiel 1:** $r \circ s \sqsubseteq r \quad s \circ s \sqsubseteq s \quad r \circ s \circ r \sqsubseteq t$
 \rightsquigarrow regulär mit Ordnung: $s \prec r \prec t$
- ▶ **Beispiel 2:** $r \circ t \circ s \sqsubseteq t$
 \rightsquigarrow nicht regulär, da Form nicht erlaubt
- ▶ **Beispiel 3:** $r \circ s \sqsubseteq s \quad s \circ r \sqsubseteq r$
 \rightsquigarrow nicht regulär, da keine entsprechende Ordnung existieren kann

Beschränkung einfacher Rollen

- ▶ Einfache Rollen in *SHOIN* = Rollen ohne transitive Unterrollen
- ▶ In *SROIQ*: Beachtung der Rolleninklusionen nötig!

Beschränkung einfacher Rollen

Einfache Rollen sind alle Rollen ...

- ▶ die nicht auf der rechten Seite einer Rolleninklusion vorkommen,
- ▶ die Inverse von anderen einfachen Rollen sind,
- ▶ die nur auf der rechten Seite von Rolleninklusionen vorkommen, bei denen links ausschließlich einfache Rollen stehen.

(Achtung: induktive Definition)

↔ nicht-einfach sind Rollen, die direkt oder indirekt von sich selbst abhängen (und deren Überrollen)

Warum ist das wichtig?

Ausdrücke $\leq n r.C$, $\geq n r.C$, $\text{Irr}(r)$, $\text{Dis}(r, s)$, $\exists r.\text{Self}$, $\neg r(a, b)$ nur für einfache Rollen r und s erlaubt!

(Grund: Sicherstellung von Entscheidbarkeit)

Überblick *SROIQ* – TBoxen

Klassenausdrücke

Klassennamen	A, B
Konjunktion	$C \sqcap D$
Disjunktion	$C \sqcup D$
Negation	$\neg C$
Existentielle Rollenrestr.	$\exists r.C$
Universelle Rollenrestr.	$\forall r.C$
Self	$\exists s.\text{Self}$
Größer-als	$\geq n s.C$
Kleiner-als	$\leq n s.C$
Nominale	$\{a\}$

TBox (Klassenaxiome)

Inklusion	$C \sqsubseteq D$
Äquivalenz	$C \equiv D$

Überblick *SROIQ* – RBoxen & ABoxen

Rollen

Rollen	r, s, t
Einfache Rollen	s, t
Universelle Rolle	u

ABox (Fakten)

Klassenzugehörigkeit	$C(a)$
Rollenbeziehung	$r(a, b)$
Neg. Rollenbeziehung	$\neg s(a, b)$
Gleichheit	$a \approx b$
Ungleichheit	$a \not\approx b$

RBox (Rollenaxiome)

Inklusion	$r_1 \sqsubseteq r_2$
Komplexe Rolleninklusionen	$r_1 \circ \dots \circ r_n \sqsubseteq r$
Transitivität	$\text{Trans}(r)$
Symmetrie	$\text{Sym}(r)$
Reflexivität	$\text{Ref}(r)$
Irreflexivität	$\text{Irr}(s)$
Disjunktheit	$\text{Dis}(s, t)$

Agenda

- ▶ Rückblick: OWL & Überblick OWL 2
- ▶ Die Beschreibungslogik *SROIQ*
- ▶ Inferenz mit *SROIQ*
- ▶ OWL 2 DL
- ▶ OWL 2 Profiles
- ▶ OWL 2 Full
- ▶ Zusammenfassung

Wie kompliziert ist *SROIQ*?

Rückblick: *SHOIN* (OWL DL) ist sehr komplex (NEXPTIME)

Beobachtung: einige Ausdrucksmittel sind nicht wirklich nötig!

- ▶ $\text{Trans}(r)$ durch $r \circ r \sqsubseteq r$ ausdrückbar
- ▶ $\text{Sym}(r)$ durch $r^- \sqsubseteq r$ ausdrückbar
- ▶ $\text{Asym}(r)$ durch $\text{Dis}(r, r^-)$ ausdrückbar
- ▶ $\text{Irr}(s)$ durch $\top \sqsubseteq \neg \exists s.\text{Self}$ ausdrückbar
- ▶ Universelle Rolle durch transitive, reflexive Überrolle aller Rollen ersetzbar (hier nicht vertieft)
- ▶ ABox durch Nominale darstellbar, z.B. $r(a, b)$ durch $\{a\} \sqsubseteq \exists r.\{b\}$

Qualifizierte Zahlenrestriktionen kaum problematisch (bekannt und implementiert)

↪ Hauptproblem Rollenaxiome (RBox)

Rolleninklusion, Sprachen, Automaten

Wie geht man mit RBoxen um?

- ▶ RBox-Regeln ähneln formalen Grammatiken
- ▶ Jede Rolle r definiert eine reguläre Sprache: die Sprache der Rollen-Ketten, aus denen r folgt
- ▶ reguläre Sprachen \equiv reguläre Ausdrücke \equiv endliche Automaten

\rightsquigarrow Ansatz: Tableauverfahren werden mit „RBox-Automaten“ erweitert

Entscheidbarkeit von *SROIQ*

Tableauverfahren von *SROIQ* zeigt **Entscheidbarkeit**

- ▶ Algorithmus hat gute Anpassungseigenschaften: ungenutzte Merkmale belasten die Abarbeitung kaum (“pay as you go”)
- ▶ Tableau-Verfahren ungeeignet für enge Komplexitätsabschätzungen
- ▶ *SROIQ* 2-NEXPTIME-komplett
 - ▶ *RIQ* and *SROIQ* are Harder than *SHOIQ*. Yevgeny Kazakov. In Gerhard Brewka and Jérôme Lang, editors, KR 2008. Pages 274-284. AAAI Press. 2008
 - ▶ Untere Schranke: Kodierung eines entsprechenden Tiling-Problems
 - ▶ Obere Schranke: Exponentielle Übersetzung in das 2-Variablen Fragment der Prädikatenlogik erster Stufe mit Zählquantoren (counting quantifiers, C_2 , Erfüllbarkeit Testen NEXPTIME-komplett)

Agenda

- ▶ Rückblick: OWL & Überblick OWL 2
- ▶ Die Beschreibungslogik *SROIQ*
- ▶ Inferenz mit *SROIQ*
- ▶ OWL 2 DL
- ▶ OWL 2 Profiles
- ▶ OWL 2 Full
- ▶ Zusammenfassung

OWL 2 DL: Weitere Aspekte

SROIQ ist “nur” logische Grundlage von OWL 2 DL

Weitere nicht-logische Aspekte:

- ▶ Syntax (Erweiterung nötig)
- ▶ Datentypdeklaration und Datentypfunktionen, neue Datentypen?
- ▶ Metamodellierung: “Punning”
- ▶ Kommentarfunktionen und ontologische Metadaten
- ▶ Invers-funktionale konkrete Rollen (Datentyp Property): Keys?
- ▶ Mechanismen zu Ontologieimport?
- ▶ ...

↪ diverse kleine Änderungen

Metamodellierung

Metamodellierung

Spezifikation ontologischen Wissens *über* einzelne Elemente der Ontologie (einschließlich Klassen, Rollen, Axiome).

Beispiele:

- ▶ “Die Klasse *Person* wurde am 30.1.2008 von *bglimm* angelegt.”
- ▶ “Für die Klasse *Stadt* wird die Property *Einwohnerzahl* empfohlen.”
- ▶ “Die Aussage ‚Dresden wurde 1206 gegründet‘ wurde maschinell ermittelt mit einer Sicherheit von 85%.”

(Vergleich auch Reifikation in RDF Schema)

Wortspiele in OWL: Punning

Metamodellierung in ausdrucksstarken Logiken ist gefährlich und teuer!

OWL 2 unterstützt zurzeit einfachste Form von Metamodellierung:

Punning

- ▶ Bezeichner für Klassen, Rollen, Individuen müssen nicht disjunkt sein
- ▶ Keine *logische* Beziehung zwischen Klasse, Individuum und Rolle gleichen Namens
- ▶ Beziehung nur relevant für pragmatische Interpretation

Beispiel:

```
Person(Birte)  klasseErstelltVon(Person, bglimm)
```


Kommentare und Metadaten

Punning unterstützt einfache Metadaten mit (schwacher) semantischer Bedeutung

Wie kann man rein syntaktische Kommentare zu einer Ontologie machen?

- ▶ Kommentare in XML-Dateien: `<!-- Kommentar -->`
↪ kein Bezug auf OWL-Axiome dieser Datei
- ▶ nicht-logische Annotationen in OWL:
`owl:AnnotationProperty`
↪ fest verknüpft mit (semantischem) ontologischem Element, kein syntaktischer Bezug

OWL 2 soll „echte“ syntaktische Kommentare unterstützen

Syntaktische Fragen

Neue/Erweiterte Syntaxen:

- ▶ RDF/XML: Erweiterung mit OWL 2 Elementen
- ▶ Funktionale Syntax (functional-style syntax): ersetzt “Abstrakte Syntax” von OWL 1
- ▶ OWL/XML: Syntax zur einfachen Verarbeitung in XML Tools
- ▶ Turtle: RDF Triple Syntax
- ▶ Manchester Syntax: Für Menschen leichter lesbare Syntax

Quo vadis, OWL Lite?

OWL Lite als Fehlschlag:

- ▶ Beinahe so komplex wie OWL DL
- ▶ Komplizierte Syntax gibt keinen direkten Zugang zu wahrer Ausdrucksstärke
- ▶ Verwendung in Ontologien heute praktisch nur „zufällig“, nicht bewusst

Ursprüngliches Ziel:

einfach und effizient implementierbarer Teil von OWL

↪ OWL 2 Profiles

Agenda

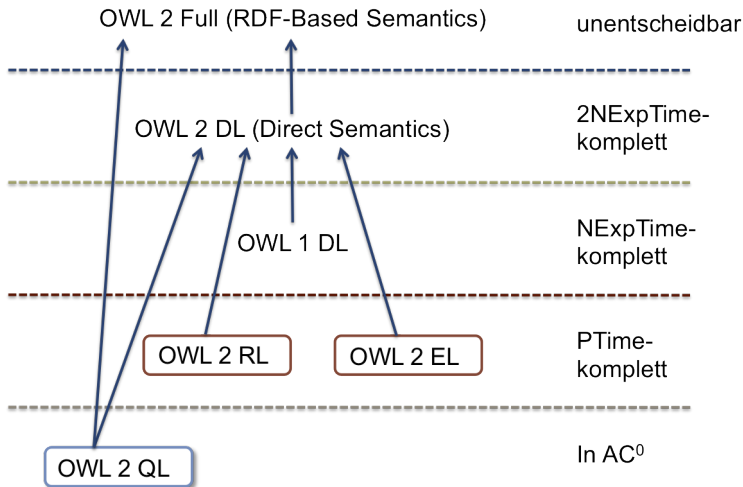
- ▶ Rückblick: OWL & Überblick OWL 2
- ▶ Die Beschreibungslogik *SROIQ*
- ▶ Inferenz mit *SROIQ*
- ▶ OWL 2 DL
- ▶ OWL 2 Profiles
- ▶ OWL 2 Full
- ▶ Zusammenfassung

OWL 2 Profiles

OWL 2 definiert drei Sprachfragmente in denen automatisches Schlussfolgern für bestimmte Aufgaben polynomiell ist

- ▶ OWL EL
 - ▶ Berechnung der Klassenhierarchie (alle Unterklassen Beziehungen) in PTIME
- ▶ OWL QL
 - ▶ Konjunctive Abfragen in $AC_0 \rightsquigarrow$ Reduzierbar zu SQL
- ▶ OWL RL
 - ▶ Kann als Erweiterung von RDFS genutzt werden oder mit der OWL Direct Semantics
 - ▶ Komplexität PTIME

OWL 2 Profiles



OWL 2 EL

- ▶ Ein (fast maximales) Fragment von OWL 2 so dass
 - ▶ Erfüllbarkeit in PTime geprüft werden kann (PTime-komplett)
 - ▶ Datenkomplexität für ABox Anfragen auch PTime-komplett
- ▶ Klassenhierarchie (alle Subsumptionsbeziehungen zwischen Klassen) können in einem Lauf berechnet werden “one pass”
- ▶ Nutzt Saturierungsverfahren das für die BL \mathcal{EL} entwickelt wurden
 - ⇒ Kann bis zum Horn (non-disjunktives) Fragment von OWL DL erweitert werden [Kazakov 2009]

OWL 2 EL

- ▶ Erlaubt:
 - ▶ Subklassen Axiome mit Konjunktion, existentiellen Restriktionen, \top , \perp , abgeschlossene Klassen mit *einem* Individuum (Nominal)
 - ▶ Komplexe Rollenaxiome, Range Restriktionen (unter bestimmten Bedingungen)
- ▶ Nicht erlaubt:
 - ▶ Negation, Disjunktion, universelle Restriktionen, inverse Rollen

OWL 2 QL

- ▶ Ein (fast maximales) Fragment von OWL 2 so dass
 - ▶ Datenkomplexität von konjunktiven Anfragen ist in AC^0
- ▶ Kann Abfragen umschreiben, so dass kein terminologisches Wissen mehr berücksichtigt werden muss (query rewriting)
 - ⇒ Standard RDBMS können zur Datenhaltung und für Abfragen verwendet werden
- ▶ Zahlreiche aktuelle Forschungsergebnisse von BLs können angewendet werden
 - ⇒ Neue Techniken zur Vermeidung einer exponentiellen Explosion beim query rewriting [Kontchakov et al. 2010, Rosati & Almatelli 2010]
 - ⇒ Kann auch auf ausdrucksstärkere Sprachen erweitert werden (-> Datalog, [Perez-Urbina et al. 2009])

OWL 2 QL

- ▶ Erlaubt:
 - ▶ Einfache Rollenhierarchien, Domain & Range Axiome
 - ▶ Subklassen Axiome mit linker Seite: Klassenname oder existentieller Restriktion mit \top , rechte Seite: Konjunktion von Klassennamen, existentielle Restriktion, und Negations von Ausdrücken, die auf der linken Seite stehen dürfen
- ▶ Unterstützt RDFS bei wohlgeformten Graphen

OWL 2 RL

- ▶ Ein (fast maximales) Fragment von OWL 2 so dass
 - ▶ Automatisches Schlussfolgern ist PTime-komplett (Konsistenz, Erfüllbarkeit von Klassen, Subsumption, Typen von Individuen prüfen und konjunktive Abfragen)
 - ▶ Automatisches Schlussfolgern ist korrekt (sound & complete) wenn der gegebene RDF Graph bestimmte Eigenschaften erfüllt
 - ▶ Andernfalls kann automatisches Schlussfolgern unvollständig sein (sound & incomplete)
- ▶ Kann direkt auf RDF Tripeln arbeiten um Instanzdaten anzureichern (Materialisierung, forward chaining für Fakten)
- ▶ Automatisches Schlussfolgern kann mit einem Satz von Regeln implementiert werden (rule engine mit Unterstützung von Gleichheit)

Agenda

- ▶ Rückblick: OWL & Überblick OWL 2
- ▶ Die Beschreibungslogik *SROIQ*
- ▶ Inferenz mit *SROIQ*
- ▶ OWL 2 DL
- ▶ OWL 2 Profiles
- ▶ **OWL 2 Full**
- ▶ Zusammenfassung

Was tun mit OWL Full?

Ziel von OWL 2 DL: viele OWL Full 1.0 Ontologien als DL interpretierbar machen (siehe z.B. Punning)

Was soll aus OWL Full 1.0 werden?

- ▶ Erweiterung von OWL Full im Sinne von OWL 2 wird durch verschiedene Anwender unterstützt
- ▶ Kann auf beliebige RDF Graphen angewendet werden
- ▶ Viele Tools in der Verifikation können auch keine Terminierung garantieren, sind aber trotzdem nützlich
- ▶ Erlaubt andere Implementierungstechniken, die evtl. schneller sind, aber keine Terminierungs-Garantien geben können

Wesentliche Unterschiede in den Semantiken

- ▶ Annotationen haben keine Semantik in der Direct Semantics, aber in der RDF-Based Semantics
- ▶ Import Anweisungen sind nur Parser Anweisungen in der Direct Semantics, aber haben eine Bedeutung als Triple in der RDF-Based Semantics
- ▶ In der RDF-Based Semantik sind Klassen Individuen, die auch noch eine Extension haben \rightsquigarrow Semantische Konditionen sind nur auf Klassen anwendbar, die auch wirklich von einem Individuum repräsentiert werden

Wesentliche Unterschiede in den Semantiken

Beispiel

- ▶ $C(a)$
- ▶ Frage nach Instanzen der Klasse $C \sqcup D$
- ▶ RDF-Based Semantics: \emptyset , Direct Semantics: a
- ↪ Unter der RDF-Based Semantik ist nur garantiert, dass die Vereinigung der Extensionen von C und D als Teilmenge der Domäne existiert; es muss kein Element existieren, dass diese Menge als Extension hat.
- ↪ In der Direct Semantics repräsentieren Klassen Mengen und keine Domänenelemente
- ↪ Antwort unter beiden Semantiken nach Erweiterung um $E \equiv C \sqcup D$

Agenda

- ▶ Rückblick: OWL & Überblick OWL 2
- ▶ Die Beschreibungslogik *SROIQ*
- ▶ Inferenz mit *SROIQ*
- ▶ OWL 2 DL
- ▶ OWL 2 Profiles
- ▶ OWL 2 Full
- ▶ Zusammenfassung

Zusammenfassung

OWL 2 als erste Weiterentwicklung des OWL-Standards

- ▶ Standardisiert am 27.10.2009
- ▶ Logische Erweiterung: Beschreibungslogik *SR_QIQ* als Grundlage
- ▶ Neue Ausdrucksmittel vor allem komplexe Rollenaxiome, qualifizierte Zahlenrestriktionen
- ▶ Nicht-logische Erweiterungen: Punning, Kommentare, Datentypen, u.a.
- ▶ Profiles mit polynomiellen Verfahren